

REMARKS

The present amendment is submitted in response to the Final Office Action mailed December 17, 2007 and telephone interview between Applicant's attorneys and the Examiner. Claims 3-15 are pending in this application. In view of the remarks to follow, reconsideration and allowance of this application are respectfully requested. Applicant appreciates the courtesy granted to Applicant's attorneys by the Examiner, during the telephone interview conducted on April 2, 2008.

Claim Objections

Claims 4-5 and 7-9 have been objected to for being dependent upon cancelled claims. Claims 4-5 and 7-9 have been amended in a manner which is believed to overcome the objections noted by the Examiner. Accordingly, withdrawal of the objection to the claims is respectfully requested.

35 U.S.C. §103(a)

In the Final Office Action, Claims 3-4 and 7-8 were rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Pat. No. 6,697,950 to Ko in view of U.S. Patent No. 4,843,545 to Kikuchi.

As per the rejection of Claim 3, the Examiner maintains Ko discloses a method for detecting malicious code patterns, the method comprising determining whether the values of tokens in two sentences in program code have the same value at the time of execution by one of: (a) determining if both tokens in the two sentences are constants and the other token is a variable, and if said determination is true, further determining whether relevant token character strings are identical to each other. The Examiner

maintains Ko allegedly teaches step (a) in Fig. 2, steps 206, 212 and Fig. 4, steps 402, 408, 410, step (b) at Fig. 2, steps 206, 212, Fig. 4, steps 402, 408, 410 and Col. 2, lines 37-47 and step (c) at Fig. 2, steps 206, 212, Fig. 4, steps 402, 408, 410 and Col. 2, lines 37-47 and Col. 4, lines 37-63. The Examiner does admit Ko fails to teach step (d) of Claim 3, however, cites Kikuchi for allegedly remedying this deficiency in Ko, citing the Abstract, Fig. 1, steps 3-6, Fig. 5, steps 17 and 19 and Col. 3, lines 29-68 in support.

Application respectfully traverses the rejection of Claim 3 for at least the reasons provided below; in this regard, independent Claim 3 has been amended herein to better define Applicant's invention over Ko and Kikuchi, alone and in any reasonable combination. Claim 3 now recites limitations and/or features which are not disclosed by Ko and Kikuchi, alone and in any reasonable combination.

Claim 3 as amended herein now recites:

3. A method for detecting malicious code patterns using a static analysis in consideration of control and data flows, the method comprising: determining whether the values of tokens in two sentences of program code have the same value at the time of execution by one of:

(a) determining, during execution, if both of the tokens of respective macro operations in the two sentences are constants for each block within a control flow graph and if said determination is true, further determining whether relevant token character strings are identical to each other;

(b) determining if one of the tokens of a macro operation in the two sentences is a constant and the other token of a respective macro operation is a variable, and if said determination is true, further determining whether the relevant token character strings are identical to each other after the variable is substituted for the constant by performing a constant propagation and if said determination is true, detecting said malicious code pattern;

(c) determining if both of the tokens in the two sentences are variables and have the same name and range, and if said determination is true, further determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof **by performing a copy propagation** and if said determination is true, detecting said malicious code pattern;

(d) determining if both of the two tokens of the two sentences are variables but do not have the same name and range, and if said determination is true, further determining whether there are definitions of the relevant variables in a control flow from a preceding one of the two sentences to a following one thereof after the relevant variables are substituted for the original variables **by performing a copy propagation** and if said further determination is true, detecting said malicious code pattern.

With respect to step (a), it is respectfully asserted Ko does not teach step (a) for at least the following reasons. Step (a) recites in part - *determining, during execution, whether both tokens **of respective macro operations** in the two sentences are constants **for each block within a control flow graph** and if said determination is true...*

With respect to step (a) of Claim 3, it is respectfully submitted Ko does not compare tokens of respective macro operation, as alleged in the Final Office Action, but instead teaches, at step 410, a step of comparing suspect macro operations against **a profile** containing information about suspect macro operations. Comparing macro

operations against a profile, as taught in Ko, is different from comparing macro operations against each other, as recited in Claim 3. Ko teaches at Col. 5, lines 10 – 30:

Intermediate form 210 feeds in analyzer 212, which performs control flow analysis and data flow analysis on the macro operations. This includes both forward data flow and backwards data flow analysis on the macro operations.

Analyzer 212 compares macro operations encountered during the flow analysis with suspect macro operations specified in profile database 214.

These suspect operations can include operations such as modifying data within another document, modifying other files in the computer system, deleting other files in the computer system, modifying operating system parameters in the computer system, exhausting a resource in the computer system, killing a process in the computer system, sending an electronic mail message to another computer system, causing a program to be run on the computer system, modifying macro operations in the document, locking a file in the computer system, and invoking a common object model (COM) object in the computer system.

In further distinction from Ko, it is respectfully submitted Ko does not compare the respective macro operations **within each block within a control flow graph**. The specification defines a basic block, at page 6, lines 20 – 24, as a series of sentences having a single entrance point and a single exit point on a control flow and becomes a node of the control flow graph. In contrast, Ko does not teach or suggest comparing

respective macro operations within each block within a control flow graph. Instead, Ko teaches in the summary, statically analyzing macro operations within a document.

The summary of Ko recites, in part, at column 2, lines 32-47:

One embodiment of the present invention provides a system that detects a macro virus in a computer system by **statically analyzing macro operations within a document**. The system operates by receiving the document containing the macro operations. **The system locates the macro operations within the document**, and performs a flow analysis on the macro operations within the document to determine associated values for variables within the macro operations. Next, the system compares the macro operations including the associated values for variables against a profile containing information about suspect macro operations and associated values for variables to determine whether the document contains suspect macro operations. If so, the system informs a user that the document contains suspect macro operations.

With respect to step (b), Applicant respectfully submits the afore-mentioned distinction, cited above with respect to step (a), wherein Ko does not compare macro operations against each other, but rather compares macro operations against a profile, also applies to step (b) of claim 3. In other words, Ko does not compare two macro operations against each other to determine if one is a constant and the other is a variable. Rather, Ko compares the respective macro operations against a profile.

Furthermore, with respect to step (b), the definition of a macro operation, as taught in Ko, is different from the definition of a macro operation, as taught in the instant application. The definition of a macro operation, as used in Ko, do not contain computer

source code. It therefore follows because Ko converts macro operations into tokens, the tokens by default do not contain computer source code. In support of this distinction, Applicant refers to Ko at Col. 4, lines 45 – 55:

The macro operations in document 108 are triggered in response to actions being performed on the document 108. For example, certain macro operations can be triggered in response to a document being opened, while other macro operations can be triggered in response to the document being closed. Also note that the term "macro operation" or macro instruction as used in this document **does not refer to preprocessing instructions for a compiler that are commonly found in computer program source code.** Instead, the term "macro operation" refers to an operations (or instruction) for an application other than a compiler that is found in a document that does not contain computer source code.

With respect to step (d), Kikuchi does not remedy the deficiency of Ko directed to allegedly teaching step (d). First and foremost, Kikuchi appears to be non-analogous art that is directed to a compile method for generating object program code from source program code in a computer. There is no teaching or suggestion of identifying malicious code. Kikuchi is directed to providing a compile method in which a variable defined by a polynomial expression including a plurality of variables can be eliminated, thereby generating object program codes of which the execution speed in an actual computer is not lowered. The mere recitation of a copy propagation operation does not qualify Kikuchi as analogous art. Moreover, copy propagation is performed, not to detect malicious code, but instead to speed up a compilation process. Further, the first condition,

relating to a polynomial expression, to determine whether a copy propagation should be performed (as recited immediately below) bears no relevance to the conditions set out in step (d) of Claim 3.

The summary of Kikuchi states, in part, at column 1, lines 57-68:
.....there is provided a predetermined condition determining whether or not the deletion of the variable can be effected and the elimination of the variable is achieved after it is judged whether a variable as an object of the deletion satisfies the condition or not. More concretely, **the first condition is that the first variable defined by a polynomial expression in the first statement is used only once by the second statement. The second condition is that there does not exist a statement changing the value of the variable between the first statement and the second statement.**

Applicant emphasizes the conditions set out in Ko for determining whether a first variable defined by a polynomial expression in a first statement is used only once by a second statement bears no relevance whatsoever to the conditions spelled out in step (d).

The Examiner also cites Kikuchi at Fig. 1, steps 3-6, which recites steps for interpreting statements of a source code program to generate intermediate code including a control flow analysis 3, a data flow analysis 4, a primary optimization processing 5 and a copy propagation 6. It is submitted none of these steps teach step (d). These steps are recited at such a general level that they cannot possibly teach or suggest the elements of step (d).

In the Final Office Action, the Examiner further asserts Kikuchi teaches the elements of step (d) at Fig. 5, steps 17 and 19. Step 17 of FIG. 5 teaches a step of checking “whether conditions are satisfied or not that the statement including the polynomial has an external procedure and that between the statement and a statement including the use point, there exists a statement including the use point, there exists a statement using an argument of the external procedure or using a variable for which ‘common’ is declared in connection with the variable.”

Determining whether a COMMON declaration is specified for a variable in a statement of an external procedure and the value the variable is changed, is starkly different from determination step (d). It therefore follows step 19 bears no relevance to step (d).

Accordingly, Applicant respectfully request withdrawal of the rejection under 35 U.S.C. §103(a) with respect to Claim 3 and allowance thereof is respectfully requested. Claims 4 and 7-8 depend from independent Claim 3 and therefore contain the limitations of Claim 3 and believed to be in condition for allowance for at least the same reasons given for Claim 1 above. Accordingly, withdrawal of the rejection under 35 U.S.C. §103(a) and allowance of Claims 4 and 7-8 is respectfully requested. .U.S. Patent Nos. 7,185,327 to Scales and 5,937,196 to Schmidt add nothing to the teachings of the other references which would render obvious the invention recited in any pending claim. Claims 10 – 15 have been added. New Claim 10 incorporates dependent claim 4 into independent form including the limitations of Claim 3 and Claims

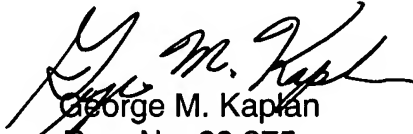
11 – 15 are dependent claims which mirror dependent Claims 5 – 9. Accordingly, there is clearly adequate support for these new claims.

Conclusion

In view of the foregoing amendment, accompanying remarks and telephone interview, it is respectfully submitted that all claims presently pending in the application, namely, Claims 3- 15 are believed to be in condition for allowance and patentably distinguishable over the art of record.

If the Examiner should have any questions concerning this communication or feels that an interview would be helpful, the Examiner is requested to contact Applicant's representative. The requisite RCE transmittal papers and filing fee, together with a Petition for one month extension of time for response under 37 C.F.R. §1.136(a) and petition fee, are enclosed.

Respectfully submitted,


George M. Kaplan
Reg. No. 28,375
Attorney for Applicants

DILWORTH & BARRESE, LLP
333 Earle Ovington Blvd.
Uniondale, NY 11553
(516) 228-8484